Exploring value prediction limits

André Seznec, Kleovoulos Kalaitzidis

Exploring value prediction limits*

André Seznec and Kleovoulos Kalaitzidis
Univ Rennes, Inria, CNRS, IRISA

Outline

In this study we explore the performance limits of value prediction for unlimited size predictors in the context of the Championship Value Prediction evaluation framework (CVP). The CVP framework assumes a processor with a large instruction window (256-entry ROB), an aggressive instruction front-end fetching 16 instructions per cycle, an unlimited number of functional units, and a large value misprediction penalty with a complete pipeline flush at commit on a value misprediction.

This framework emphasizes two major difficulties that an effective hardware implementation value prediction will face. First the prediction of a value should be forwared to the pipeline only when the potential performance benefit on a correct prediction outweigths the potential performance loss on a misprediction. Second, value prediction has to be used in the context of an out-of-order execution processor with a large instruction window. In many cases the result of an instruction has to be predicted while one or several occurrences of the same instruction are still progressing speculatively in the pipeline. In this study, we illustrate that these speculative values are not required to deliver state-of-the-art value prediction.

Our proposition ES-HC-VS-VT combines four predictor components which do not use the speculative results of the inflight occurrences of the instruction to compute the prediction. The four components are respectively the E-stride predictor(ES) [13], the HCVP, Heterogeneous-Context Value, predictor (HC)[9], the VSEP, Value Speculative Equality Predictor (VS) [3] and the VTAGE predictor (VT) [6]. Prediction is computed as, first VTAGE prediction, if not high confidence: VSEP prediction, if not high confidence: HCVP prediction, if not high confidence: E-stride prediction. E-Stride computes its prediction from the last committed occurrence of the instruction and the number of speculative inflight occurrences of the instruction in the pipeline. HCVP computes the predicted value through two successive contexts; first the PC and the global history are used to read a stride history, then this stride history is used to obtain a value and a stride. On VTAGE and VSEP, the predicted value is the value directly read at prediction time on the predictor tables.

As for EVES [13], for ES-HC-VS-VT, we optimize the algorithms to assign confidence to predictions on each predictor component depending on the expected benefit/loss of a prediction.

On the 135 traces from CVP, ES-HC-VS-VT achieves **4.030 IPC** against **3.881 IPC** achieved by the previous leader of the CVP, HCVP+E-stride [9].

*This work was partially supported by an Intel research grant

1 The ES-HC-VS-VT predictor

The predictor consists in 4 predictor components, respectively the E-stride predictor (ES) [13], the HCVP, Heterogeneous-Context Value predictor (HC) [9], the VSEP [3], Value Speculative Equality Predictor (VS) and the VTAGE predictor (VT) [6] that are all checked in parallel. If any of the components provide a prediction with high confidence, the prediction is used in the pipeline. If two or more components provide predictions with high confidence then the priority is first VTAGE, then VSEP, then HCVP, then E-stride. The different components are described below.

1.1 The Enhanced Stride Predictor

The E-stride predictor is derived from the stride predictor [5] and was introduced in [13]. The stride predictor [5] computes the predicted value as the result of the addition of the last result of the instruction and a stride. The stride is dynamically computed and updated at validation time. The stride is read on the predictor table while the last value is read either on the predictor table or when there is at least an inflight occurrence of the instruction in the pipeline from the speculative instruction window.

The prediction is only used in the pipeline when both the stride and the last value are high confidence. That is if any of the inflight occurrences present in the pipeline was not high confidence then one can not use the prediction: when the stride becomes high confident for instruction I, one cannot begin using the hardware value predictions before all the occurrences of instruction I have disappeared

from the instruction window. On loops, this might never happen unless the pipeline is flushed. In order to avoid this issue, E-stride implements a modified version of the stride prediction algorithm. The predicted value for instruction I is computed as the Last **Committed** Value added with $(Inflight + 1)$*Stride where $Inflight$ is the number of speculative occurrences of instruction I. As soon as the stride associated with instruction I reaches high confidence, the prediction can be used even if other (not predicted) occurrences of the same instruction are still present in the pipeline. The same scheme was suggested to use value prediction for predicting branches [2].

1.2 The Heterogeneous Context Value Predictor

The Heterogeneous Context Value Predictor, HCVP, [9] is a differential value predictor as DFCM [1], i.e. it computes the prediction of an instruction result as the sum of the last result encountered in the same context and a stride which is computed as the difference between the two last results encountered in the same context. Other predictors from this family are DFCM [1] and DVTAGE [7].

HCVP features a three steps prediction algorithm. First, the instruction address and the global branch history are used to retrieve a stride history. Then this stride history is combined with the instruction address and the global branch history to retrieve two values, a stride and the last value associated with this context. Finally, the stride and last value are added to compute the predicted value.

One difficulty for a potential hardware implementation of HCVP is that the same context can appear in the instruction window, thus requiring the use of the speculative value and the speculative stride history for the prediction computation. However when using a very long branch history - more than 100 branches-, the probability that the same context (instruction address and global branch history) occurs along a 256-instruction window is very low. Therefore in the current implementation of HCVP, if a context is already inflight in a speculative instruction, no prediction is done in the simulation [9].

1.3 VTAGE

VTAGE was introduced in [6] and is directly derived from the indirect branch predictor ITTAGE [12]. VTAGE uses the PC and the branch history to predict the instruction result. VTAGE uses several tables for storing predictions. Each table is indexed by a different number of bits of the global branch history, hashed with the PC of the instruction. The different lengths form a geometric series [12]. The tables are backed up by a base predictor – a tagless Last Value Predictor [4]– which is accessed using the instruction address only. A contrario from the initial VTAGE proposition, we implement a 2-way skewed associative tagged base predictor. In VTAGE, an entry of a tagged component consists of a partial tag, a usefulness counter *u* used by the replacement policy, a full 64-bit value *val*, and a confidence counter c. At prediction time, all components in VTAGE are searched in parallel to check for a tag match. The matching component accessed with the longest history provides the prediction and its confidence to the pipeline.

1.4 Value Speculative Execution Predictor

Context-based value predictors [10] aim to capture the repetition of the same value in the results of a static instruction when using a specific context, e.g. the same sequence of results or the same global branch history. For instance, VTAGE [6] leverages the execution's global branch history, i.e., the same value encountered with the same global branch history. VSEP [3] aims at capturing another form of value regularity: equal values on consecutive occurrences of the same static instruction. In particular, it aims at capturing interval equality; i.e., the cases where a static instruction produces the same result on an interval, but for each interval the result is different. The challenge is to determine interval interruptions to avoid mispredictions. Typically VTAGE often captures this interval interruption, but suffers from cold entries, i.e., low confidence at the beginning of each new interval. Rather than predicting the value for a given global branch history, VSEP predicts if the result of the current occurrence is equal to the last committed occurence of the instruction.

VSEP consists in two distinct components, ETAGE, the equality predictor, and LCVT, the Last Committed Value Table. ETAGE is a context-based equality predictor that essentially copies the TAGE branch predictor structure. ETAGE predicts *equality* or *inequality* between the value to be produced by the current instance of a static instruction and the **last committed value** of the same instruction. LCVT records all the committed values.

Typically if the beginning of the interval equality is strongly correlated with the global branch history, VSEP will be able to predict the new value once the first occurrence in the new interval has been committed, i.e. flown out from the instruction window, i.e after a few occurrences in the interval.

2 Tailoring the confidence assignment algorithms

All value predictions are not equal. Only values predicted with a high confidence are forwarded to the pipeline. However predicting the result of a load missing the LLC cache is more likely to lead to a performance benefit than predicting the result of a single cycle ALU operation. On

the other hand, a large penalty is paid on every misprediction. That is, while one can tolerate misprediction rates of 1% or even higher on predictions for loads that are likely to miss through the whole hierarchy, the misprediction rate on single cycle ALU operations should be close to zero. Therefore, reaching high confidence should be easier in the former case than in the latter case.

As already done for EVES [13], we have designed confidence assignment algorithms that discriminate between instructions based on their actual latency.

Our algorithms to manage confidence use multi-bit probabilistic counters [8]. For each of the predictor components, we carefully define the probabilities of incrementing the confidence counter depending on the instruction type, the instruction latency and the stride or the predicted value.

2.0.1 E-stride

In our submission, the probability of incrementing the counter on a load is 1 on a miss on the LLC, $\frac{3}{4}$ on a hit on the LLC, $\frac{5}{16}$ on a hit on the L2 cache, $\frac{9}{64}$ on a hit on the L1 cache. For a slow ALU or a floating point operation, probability $\frac{1}{32}$ is used. For single cycle ALU instructions, probability $\frac{1}{128}$ is used. Finally we remarked that small strides (e.g. 1) do not bring the same benefit as large ones; therefore for small strides, the confidence counter is incremented with a lower probability.

Abruptly resetting the confidence counter on a misprediction often leads to successions of warming periods with no benefits from stride predictions followed by a period of useful predictions. Therefore we opted for a decrease of the confidence counter by 4. With a 5-bit counter and a high confidence threshold of 7, in most cases the use of the prediction can restart immediately after a misprediction.

2.0.2 HCVP

In our submission, we also adapt the probability of incrementing the confidence counter on HCVP to the potential benefit of predicting the instruction. Typically, loads missing the LLC will always increment the confidence counter while an ALU instruction will increment the confidence counter with probability $\frac{1}{16}$. Moreover as for E-stride, for small strides, particularly null stride, we further decrease this probability.

The confidence counter is reset on a misprediction. The threshold for prediction use was fixed to 16. We found that such a threshold decreases the number of mispredictions, but at some cost on the coverage. To limit this impact, when an entry is allocated, its confidence is set to 12; after a first misprediction (generally before reaching the 16 threshold), the entry becomes unlikely to regain high confidence.

2.0.3 VSEP and VTAGE

In this submission, the confidence updating algorithm for VTAGE and VSEP are directly derived from the one implemented in [13]. Probabilities were slightly adjusted to adapt to the inclusion in a 4-components value predictor instead of a 2-components predictor. In this submission, we use 4-bit confidence counters instead of 3-bit counters. On a correct prediction, the probability of increasing the confidence varies from 1 for a load missing the whole hierarchy to $\frac{1}{256}$ for a single cycle ALU instruction producing a null value.

3 Miscelleneaous Optimizations

Several traces were encountering slowdowns with the E-Stride predictor. In order to avoid/decrease this phenomenon, we implemented a simple safety net. The ratio of misprediction associated with E-stride on the ratio of instructions is monitored and maintained under $\frac{1}{1024}$ through a 16-bit counter (SafeStride in the code) (see [13]).

As for EVES [13], to avoid burst of mispredictions on VTAGE, HCVP, and VSEP, a high confidence prediction on a component is not used when a misprediction has occurred in the last 256 (128 for VSEP) instructions. This has a small performance impact, but significantly limits the number of mispredictions.

On some benchmark traces, VTAGE, HCVP and VSEP are encountering much higher misprediction rates than on other benchmarks. To limit this phenomenon, we implement a form of dynamic threshold fitting [11], limiting the misprediction rate to a maximum of $\frac{1}{128}$ th on HCVP and $\frac{1}{512}$ th on VTAGE and VSEP. This optimization slightly limits the misprediction rate and slightly improves the performance (overall +0.018 IPC).

4 Performance evaluation

The performance of the ES-HC-VS-VT predictor is evaluated as the geometric mean of the performance of the 135 traces of the CVP. The global performance obtained is **4.030** IPC to be contrasted with the performance of the previous leader E-Stride+HCVP, **3.881** IPC [9] and the performance of the initial EVES proposition at CVP1 **3.773** IPC [13]. ES-HC-VS-VT implements both EVES (disabling HCVP and VSEP) and E-Stride+HCVP (disabling VSEP and VTAGE). Optimizations that were brought to the design brings the respective performances to **3.785** IPC for EVES and **3.924** IPC for HCVP.

We further analyse the performance of each component in the predictor. First by disabling the use of one component in the predictor, to illustrate its unique contribution, i.e. the benefit it brings on top of the other components. Orthogonally we also illustrate the performance brought by

	ES-HC-VS-VT	no ES	no HC	no VS	no VT	ES	HC	VS	VT
IPC	4.030	3.600	3.827	4.012	3.981	3.183	3.511	2.980	3.257
Coverage (%)	63.9	61.1	48.4	62.4	58.1	3.6	49.8	27.8	42.5
misp. rate (%)	0.061	0.041	0.055	0.061	0.062	0.6	0.042	0.018	0.015

Table 1. Performance of ES-HC-VS-VT, when disabling one component, and when enabling a single component

	ES-HC-VS-VT	ES	HC	VS	VT
Coverage (%)	63.9	2.8	15.2	3.5	42.5
misp. rate (%)	0.062	0.51	0.115	0.059	0.015

Table 2. Performance of each component in ES-HC-VS-VT

the component alone. Table 1 illustrates the performance, the coverage (percentage of high confidence prediction), and their misprediction rates (percentage of misprediction on high confidence prediction) of the different components.

The simulation results clearly point out the importance of the E-stride component. When disabling this component, the performance loss is very important despite its coverage is quite limited and its misprediction rate is quite high (0.6 %, i.e. 15 times higher than HCVP and 40 times higher than VTAGE). On the other hand HCVP appears as the component with the highest coverage, but many of the high confidence predictions by HCVP are also high confidence for VTAGE or VSEP. VSEP appears as only bringing marginal performance over the three other components.

Table 2 illustrates the repartitions of the high confidence predictions among the different components on ES-HC-VS-VT and their misprediction rates. A prediction is associated to a component when the component was the highest priority component delivering the prediction. About $\frac{1}{4}$ of the predictions done by E-stride are also high confidence by another component. HCVP delivers high prediction on 15.9 % of the microoperations that were not addressed by VTAGE and VSEP, while VSEP delivers only limited extra coverage over VTAGE (3.5 %). It also appears that most of the mispredictions on high confidence predictions on HCVP were encountered on microoperations that were not predicted by VSEP or VTAGE.

5 Summary

This study explore the limits of value prediction with unlimited storage budgets. We have combined 4 predictor components with different characteristics, the E-stride predictor, the HCVP predictor, the VSEP predictor and the VTAGE predictor. None of these predictors uses the speculative results of inflight occurrences of instructions. E-stride and HCVP computes their prediction as the sum of a stride and a base, that is they are able to predict values that have not been already encountered. VTAGE and VSEP only predict values that are stored in their tables. Among the predictors, HCVP, VSEP and VTAGE are using global branch/path history as a context to predict. HCVP further uses the history of values computed in the context of a given global branch history.

For all these predictor components, we have proposed an efficient approach for managing confidence. We differentiate the probabilities of updating confidence counters depending on instruction type, instruction latency and instruction result value. On an effective hardware implementation, these probabilities will have to be carefully designed, depending on the management of mispredictions (at execution or at commit), the effective characteristics of the pipeline (execution width, ...), the branch predictor behavior, the criticality of the instruction, the memory level parallelism, ...

Further explorations might include adding components that necessitate to use speculative values that should be extracted from the instruction window.

References

[1] Bart Goeman, Hans Vandierendonck, and Koenraad De Bosschere. Differential FCM: increasing value prediction accuracy by improving table usage efficiency. In *Proceedings of the Seventh International Symposium on High-Performance Computer Architecture (HPCA'01), Nuevo Leone, Mexico, January 20-24, 2001*, pages 207–216. IEEE Computer Society, 2001.

[2] Timothy H. Heil, Zak Smith, and James E. Smith. Improving branch predictors by correlating on data val-

ues. In *MICRO*, 1999.

[3] Kleovoulos Kalaitzidis and André Seznec. Value Speculation through Equality Prediction. In *ICCD 2019 - 37th IEEE International Conference on Computer Design*, pages 1–4, Abu Dhabi, United Arab Emirates, November 2019. IEEE.

[4] M.H. Lipasti and J.P. Shen. Exceeding the dataflow limit via value prediction. In *Proceedings of the Annual International Symposium on Microarchitecture*, pages 226–237. IEEE Computer Society, 1996.

[5] A. Mendelson and F. Gabbay. Speculative execution based on value prediction. Technical Report TR1080, Technion-Israel Institute of Technology, 1997.

[6] Arthur Perais and André Seznec. Practical data value speculation for future high-end processors. pages 428–439, 02 2014.

[7] Arthur Perais and André Seznec. Bebop: A cost effective predictor infrastructure for superscalar value prediction. *2015 IEEE 21st International Symposium on High Performance Computer Architecture, HPCA 2015*, 02 2015.

[8] N. Riley and C. B. Zilles. Probabilistic counter updates for predictor hysteresis and stratification. In *Proceedings of the International Symposium on High Performance Computer Architecture*, pages 110–120, 2006.

[9] Chirag Sakhuja, Anjana Subramanian, Pawanbalakri Joshi, Akanksha Jain, and Calvin Lin. Combining branch history and value history for improved value prediction. In *CVP - Championship Value Prediction*, November 2019.

[10] Y. Sazeides and J.E. Smith. The predictability of data values. In *Proceedings of the Annual International Symposium on Microarchitecture*, pages 248–258. IEEE, 1997.

[11] A. Seznec. Analysis of the o-geometric history length branch predictor. volume 33, pages 394– 405, 07 2005.

[12] A. Seznec and P. Michaud. A case for (partially) tagged geometric history length branch prediction. *Journal of Instruction Level Parallelism*, 8:1–23, 2006.

[13] André Seznec. Exploring value prediction with the EVES predictor. In *CVP-1 2018 - 1st Championship Value Prediction*, pages 1–6, Los Angeles, United States, June 2018.